유전자 프로그램의 진화를 이용한 자율이동로봇의 행동 학습

이광주
서울대학교 인지과학 협동과정
kjlee@scai.snu.ac.kr

장병탁
서울대학교 컴퓨터공학부 및 인지과학 협동과정
btzhang@scai.snu.ac.kr

Learning Robot Behaviors by Evolving Genetic Programs

Kwang-Ju Lee
Interdisciplinary Program in Cognitive Science

Byoung-Tak Zhang
School of Computer Science and Engineering

Seoul National University

요 약

주어진 환경에 대한 특별한 사전 지식 없이 그 환경에 적응할 수 있는 자율이동로봇을 설계할 때는 우선 특정한 상황에서만 유효한 가정들을 될 수 있는 대로 배제하여야 한다. 본 논문에서는 이러한 적응 능력을 갖춘 자율이동로봇을 설계하기 위한 일환으로 유전자 프로그램을 이용하여 로봇의 제어기를 표현하고, 이를 진화하여 로봇이 현재 자신의 주변에서 얻을 수 있는 정보에만 기초하여 목표물을 찾아가는 행동 규칙을 학습하도록 하였다. 로봇은 현재 자신이 놓여있는 환경에 대한 지도를 작성하지 않은 채 현재 자신의 주변에서 얻을 수 있는 지역적인 정보만으로 특정 목표물을 찾아가도록 학습된다. 로봇은 먼저 단층 퍼셉트론을 사용하여 주어진 공간내의 장애물과 목표물을 인지하도록 학습된다. 그 이후 학습된 퍼셉트론을 유전자 프로그램의 함수 노드로 사용하여 트리를 진화시켰다. Khepera 시뮬레이터를 이용한 실험 결과, 로봇은 제한된 지역 정보만을 사용하여 목표물을 찾아가는 행동 규칙을 매우 안정적으로 학습할 수 있었다.

1. 서론

자율 이동 로봇이 실세계에 응용되기 위해서는 불확정적으로 변화하는 환경에서의 적응 능력이 필요하다. 이를 위해 영역 특정적인 지식이나 외부의 명시적 제어 없이 주어진 환경에 적응하는 특성을 가지는 자율 이동 로봇을 개발하려는 시도가 있어왔다[1,4,5]. 불확정적으로 변화하는 환경에서 보다 일반적인 적응 능력을 보장받기 위해서는 구체적인 가정들을 될 수 있는 대로 배제하는 것이 바람직하다. 가정들이 구체적일수록 특정한 상황에서만 유용하고 상황이 변하면 별 가치가 없어지기 쉽기 때문이다. 여기에는 주어진 환경에 대한 지도 및 패턴화된 행동 전략 등이 포함된다.

이처럼 좀더 일반적인 상황에서 적응 가능한 온라인 행동 전략 학습을 위해 진화 알고리즘을 로봇 제어에 응용한 연구들이 있다. Koza는 유전자 프로그래밍(Genetic Programming)을 이용하여 로봇이 주변 정보만으로 Grid 공간 내를 돌아다니면서 산재해 있는 먹이를 먹는 행동 규칙을 찾아 낼 수 있음을 보였다[2]. 이 연구에서 트리의 함수 노드는 IF-FOOD-HERE, IF-CARRYING-FOOD이나 PICK-UP, MOVE-RANDOM 등의 기호 표현이 사용되었는데, 이러한 노드를 실제 자율 이동 로봇에 적용하기 위해서는 각 노드들을 어떻게 구현하는가 하는 문제가 남는다.

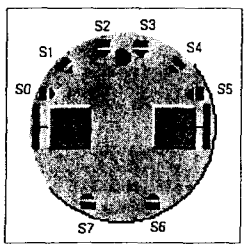
D. Floreano 등은 유전자 알고리즘을 이용하여 센서정보를 바로 입력 노드로 하고 모터 출력을 출력 노드로 하는 Recurrent Neural Network을 진화시켜 이를 실제 로봇을 제어하는데 사용하였다[1]. 이들은 Khepera Robot을 이용하여 실험하였는데, 제한된 센서 정보와 매우 단순한 Fitness function 구성으로도 밧데리가 방전되면 충전가능지역으로 찾아가는 행동 전략을 로봇 스스로가 찾아 낼 수 있음을 보였다. 그러나 이보다 좀더 복잡한 환경, 가령 장애물이 한두 개 추가되는 경우에는 이와 같은 행동을 안정적으로 찾아내지 못하였으며 진화에 소요되는 시간도 10일 정도로 매우 길었다.

본 논문에서는 일단 로봇이 센서 정보로부터 현재 자신의 상태를 판단할 수 있는 퍼셉트론(Perceptron)을 학습하여 기호 수준의 함수집합을 구축하고, 구축된 함수집합을 기반으로 하는 로봇의 제어기를 유전자 프로그램을 이용하여 진화시킴으로써 로봇이 특정 임무를 수행할 수 있는 행동 규칙을 효과적으로 학습할 수 있음을 보였다.로봇은 전체 환경에 대한 지도를 사용하지 않고 8개의 거리 측정용 센서와 8개의 광센서를 통해 자신의 주변 정보만을 사용하여 장애물을 피하면서 지정된 목표물에 다가가는 행동 규칙을 학습하여야 한다. 여기서는 Khepera 시뮬레이터[6]를 이용하여 Grid 공간이 아닌 Real-Value공간에서 이러한 행동이 진화될 수 있음을 보인다.

2. 실험 방법

A. Simulated Robot

Khepera 시뮬레이터의 로봇에는 다음 그림1과 같이 8개의 적외선 센서(Infrared Sensor, IR)와 2개의 모터가 구현되어 있는데, 8개의 적외선 센서는 적외선 반사에 의해 센서로부터 약 5cm(simulated distance) 전방의 물체 유무를 0~1023까지 수치로 나타내준다. 또한 각 IR 센서는 빛의 세기를 측정할 수 있는 광센서로도 작동할 수 있는데, 주변 빛의 세기에 따라 0~525까지의 값을 갖는다.



<그림 1>
Khepera Simulator내의 Khepera robot.

로봇의 행동은 좌/우 두 개의 모터를 제어함으로써 이루어지며, 시뮬레이션 환경에서 모터는 +10(전진)에서-10(후진)까지의 실수값을 설정함으로써 제어할 수 있다.[1)]

B. 로봇 제어 구조 및 학습

로봇 제어 유전자 프로그램에 사용된 함수 집합과 종단 집합은 표 1에 보이는 바와 같다. Function 집합 중 PROG2를 제외한 나머지 노드는 모두 각각 하나의 단층 퍼셉트론으로 구성된다.

각 퍼셉트론은 16개의 입력 단과 1개의 출력 단을 가지며 입력 단은 모두 출력 단과 연결된다. 입력 단중 8는 위 그림 1의 각 방향 IR 센서(S0~S7)의 거리 측정값과 연결되고 나머지 8개의 입력 단은 광센서(L0~L7)측정값과 연결된다. 각 입력은 센서값을 0.0~1.0 사이의 실수로 변환한 값이다. 출력 단의 출력 값과 Weight 갱신은 다음 식 (1), (2)와 같이 계산되었다.

$$o(\vec{x}) = \vec{w} \cdot \vec{x} \qquad (1)$$

$$\triangle \vec{w} = \eta\,(t-o)\vec{x} \qquad (2)$$

(단, o는 출력, $\vec{w}$는 weight vector, $\vec{x}$는 입력 vector, t는 목표값, η는 learning ratio(= 0.1))

로봇은 각 퍼셉트론을 먼저 학습하였다. 학습에 사용된 데이터는 각각의 퍼셉트론이 학습하여야 하는 상황(즉, if-obstacle-left, if-target-right 등)들을 시뮬레이션 환경에 만들고 그 안에서 각 1000여 개의 학습 표본을 추출하여 학습하였다. 이 때 목표값은 그 상황이 참이면 1.0, 거짓이면 -1.0으로 설정하였다.

퍼셉트론이 학습된 후 이 퍼셉트론을 함수 노드로 활용하여 유전자 프로그램을 진화시켰다. 이 유전자 프로그램에서 하나의 개체는 최대 깊이 10의 트리로 표현하였다. 트리 개체의 예를 그림 2에 나타내었다.

각 개체는 개체가 표현하는 행동 규칙에 따라 고정된 스텝(step)을 움직인 뒤 다음 식(3)에 의해 적합도를 계산한다.

$$F(i) = w_1 \frac{C_i}{S} + \frac{(S - w_2 H_i)}{S} \qquad (3)$$

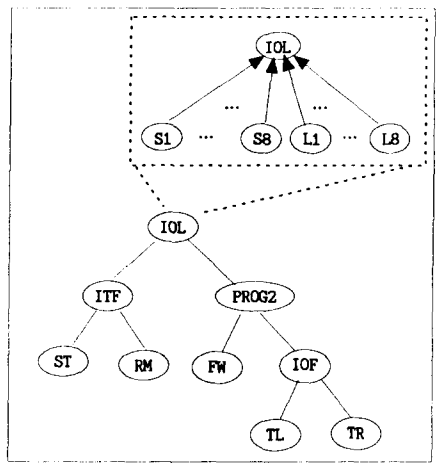
(단, S는 한 생명 주기동안의 스텝수(=2000), Ci는 충돌 횟수, Hi는 목표물에 접근한 횟수, w_1, w_2등은 weight $(w_1 = 1.0, w_2 = S/10)$)

개체의 적합도는 충돌 횟수, 목표물에 접근한 횟수 만으로으로 평가되는데, 두 가지 항목 모두 로봇에 갖춰진 제한된 센서로 판단할 수 있는 것이다.

모든 개체가 평가된 후 한 세대 중 50%의 개체를 uniform ranging selection에 의해 선택한 후 선택된 개체에 대해 Reproduction, Crossover 및 Mutation 연산을

구분	기호	의미
Function Node	IOL, IOR, IOF, IOB	장애물 존재 여부 판단 (If Obstacle is located Left/Right/Forward/Back)
	ITL, ITR, ITF, ITB	목표물 존재 여부 판단 (If Target is located Left/Right/Forward/Back)
	PROG2	왼쪽 sub tree 수행 후 오른쪽 sub tree 수행
Terminal Node	TL, TR	Turn Left/Right
	MF	Move Forward
	RM	Random Move
	ST	Stop

<표 1> 트리 구성에 사용되는 Function 및 Terminal 집합



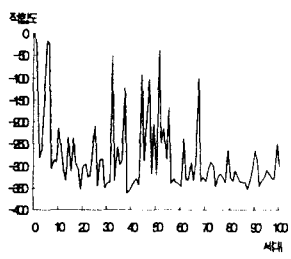
<그림 2> 로봇 제어기의 표현 예. "If-Obstacle-Left가 참이면 If-Obstacle-Forward가 참인지 보고 이것이 참이면 Stop, 아니면 Random-Move, If-Obstacle-Left가 거짓이면 Forward 한 후(PROG2) If-Obstacle-Forward가 참인지 보고 참이면 Turn-Left, 거짓이면 Turn-Right".

1) 시뮬레이션 되는 IR 센서의 반환값에는 +/-10%의 오차가 임의로 추가되며, 모터 제어의 경우 모터 속도에는 +/-10%, 방향에는 +/-5%의 오차가 임의로 추가된다.

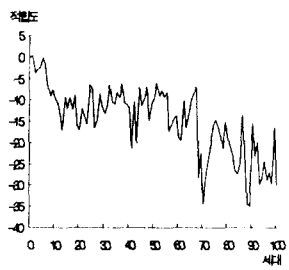
적용하여 다음 세대를 구성하였다. 이 때 사용한 Reproduction, Crossover, Mutation 비율은 각각 0.1, 0.8, 0.1 이었고, 각 세대간 개체 수는 100으로 유지하였고, 한 세대의 최적 개체는 다음 세대에서도 그대로 유지하였다.

3. 실험 결과

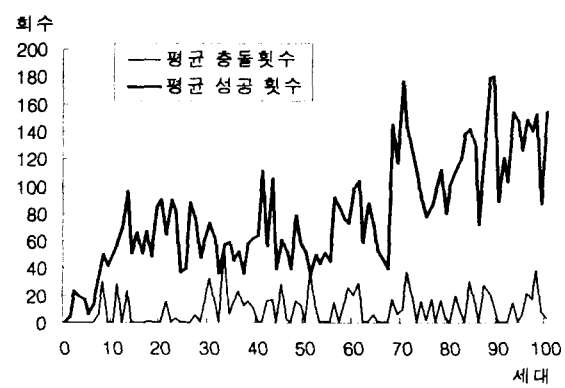
다음 그림 3, 4에 처음 100세대 동안 평균 적합도와 최적 적합도의 변화를 나타내었다. 세대별 로봇의 충돌 횟수와 목표물 근접 회수의 변화는 그림 5에 나타내었다. 그림에서 보이는 바와 같이 평균 적합도와 최적 적합도 모두 세대를 거듭할수록 향상된다. 최적 적합도의 경우 약 10세대부터 매우 낮은 값을 보이지만 편차가 심하다가 약 70세대 이후 편차가 줄어들면서 안정화 되고 있다. 최적 적합도의 편차가 비교적 크게 나타나는 것은 적합도의 평가가 개체 자체로 결정되는 것이 아니라 개체의 수행 결과로 평가되기 때문에 비록 동일한 개체로 적합도를 평가한다 하더라도 매 수행마다 값이 달라질 수 있기 때문이다. 그러나 약 70세대이후 대부분의 개체들이 일정한 상태로 수렴하면서 최적 적합도의 편차 역시 줄어들게 된다. 평균 성공 횟수는 지속적으로 증가하지만 평균 충돌 횟수는 비슷한 수준을 유지하고 있다. 하나의 개체로 2000 step 만큼 로봇을 움직이고, 전체 step 동안 충돌 횟수와 목표물에 근접한 횟수에 기반하여 적합도를 계산하였다. 이는 장애물 충돌 수 대비 성공횟수가 지속적으로 향상되고 있다는 것을 말해준다.



<그림 3> 최적 적합도의 변화



<그림 4> 평균 적합도의 변화



<그림 4> 평균 충돌 횟수와 평균 성공 횟수의 변화

4. 결론

본 논문에서는 유전자 프로그램의 진화를 통해 자율이동 로봇이 주어진 공간에 대한 지도 없이 로봇의 주변 지역 정보만으로 주어진 공간을 탐색하면서 특정 목표물을 찾아갈 수 있는 행동 규칙을 안정적으로 학습할 수 있음을 보였다.

실험은 비록 시뮬레이션 상에서 이루어졌지만 Grid 공간이 아닌 실수 공간을 사용하였고, 비교적 실제 로봇의 수행 환경과 유사한 시뮬레이션 환경 하에서 실험을 수행하였고, 시뮬레이션 상에서만 유효한 정보들은 일체 사용하지 않았기 때문에 결과를 바로 실제 로봇으로 구현하는데 많은 도움을 줄 수 있을 것이라 생각된다. 실제로 사용한 시뮬레이터는 실제 Khepera 제어 프로그램을 시험해 보기 위한 용도로 사용되는 것이기 때문에 본 실험 결과를 Khepera에 그대로 적용할 수 있을 것이다. 따라서 추후 과제로서 본 실험 결과를 실제 Khepera에 적용하여 결과를 확인해 보는 일을 수행하여야 할 것이다. 또한 현 실험에서는 Function node를 몇 가지 가능한 판단 기준을 사전에 정의하여 고정하였는데, 이후에는 주어진 환경에서 필요한 Function node 자체를 학습할 수 있도록 하는 방안을 고려하여야 할 것이다.

감사의 글

본 논문은 과학재단 핵심전문 연구(과제 번호 981-0920-107-2) 와 특정기초 연구(과제 번호 96-0102-13-01-3)에 의해 지원되었음.

참고 문헌

[1] Floreano, D. and Mondada, F. 1996. Evolution of homing navigation in a real mobile robot, *IEEE Transactions on System, Man and Cybernetics,* 26(3): 396-407.

[2] Koza, J. R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, MA: The MIT Press.

[3] Zhang, B. T. and Cho, D. Y. 1998. Coevolutionary Fitness Switching: Learning Complex Collective Behaviors Using Genetic Programming. *Advances in Genetic Programming III.* Cambridge, MA: The MIT Press. pp. 425-445.

[4] Brooks, R. A. 1991. Intelligence Without Representation. *Artificial Intelligence,* 47: 135-159.

[5] Chown, E. 1999. Making Predictions in an Uncertain World: Environmental Structure and Cognitive Maps. *Adaptive Behavior,* 7(1): 17-33

[6] Oliver Michel. Khepera Simulator Package version 2.0: Freeware mobile robot simulator written by Oliver Michel. http://diwww.epfl.ch/lami/team/michel/khep-sim/index.html